5
DNA Computation

Transparent forms, too fine for mortal sight, Their fluid bodies half dissolved in light.

—Alexander Pope, "The Rape of the Lock"

Our genetic material, or DNA, encodes information in the form of a particular sequence of molecules, and is often compared to computer software. The information it contains, which plays a major role in determining our hereditary traits, gets "processed," so to speak, through biochemical means in order to control the synthesis of proteins within the cell. The fact that genes are a means for encoding and processing information is at the crux of the idea behind DNA computation.

DNA is made up of four different kinds of molecules called nucleotides, which are organic compounds composed of a nitrogen-containing base, a phosphate group, and a sugar molecule. Their particular sequence or way they are ordered represents a code that spells out the biochemical instructions for the production of a given protein, just as a computer software program encodes instructions based on a pattern of zeros and ones.

The mode in which a segment of the DNA code gets activated to produce a specific protein depends on any one of a series of complex biochemical operations that serve to splice, copy, insert, or delete the segment. These operations are analogous to the arithmetical and logical operations performed on the zeros and ones contained in a computer program.

DNA computation makes use of this ability to store and manipulate the information in a genetic code to devise algorithms for solving a problem, in some ways even improving on what a conventional

computer could do for a certain small group of problems, claim some. Not even a decade old, DNA computation has grown considerably as an interdisciplinary field of research since the first conference in 1995, and has given rise to some interesting new avenues of research such as the self-assembly of DNA molecules, which could prove a useful tool for building structures for nanotechnology. DNA also has appeal as a storage medium—for example, a cubic centimeter of DNA stores about 10^{21} bits of information. It can perform trillions of operations in parallel—far more than any electronic computer—and uses about 10^{-9} of the energy of conventional computers.

However, DNA computing is not without its limitations, and will likely never function for general purpose computations (in other words, it's doubtful that in thirty years' time, we'll be using vials of DNA on our desktops to compute the solutions to problems). Although, as a method, it has been successful at solving small-scale search problems using combinatorial algorithms, it has been unable to scale up for larger problems, as such exponentially large amounts of DNA would be needed that the computational techniques wouldn't hold up. In addition, DNA computing even for medium-scale problems requires very painstaking laboratory work to implement, and as a consequence, usually entails a significant amount of error. Whether or not it will find its "killer ap"—the application for which it's the best means possible, transforming DNA computation into the world's next revolutionary computing device or a promising new information storage medium—is still uncertain. Some critics have also questioned the method's practicality because of the highly laborious lab work it entails and the high cost of some of the materials required.

That said, DNA computing has broken completely new ground in science—an rare achievement in itself—in addition to having spun off new research areas such as DNA self-assembly, and attracted many talented researchers to the field. Its cross-disciplinarity has lured computer scientists into the biology lab, where they have had to master the techniques of recombinant DNA, while providing their biologist colleagues with a synergistic perspective on the computational properties of living cells, introducing them to concepts such as networking, systems, input and output, switch, and oscillator that can be applied to cellular functions.

Genes, Nucleotides, and Computing

The work of mathematician Tom Head at SUNY Binghamton in the late 1980s served as an important precursor for the discovery of DNA computing. Head devised a way to use DNA sequences as a code for composing and executing algorithms. He developed a *splicing model*, experimenting with recombinant DNA lab techniques on DNA strands to carry out simple rules taken from formal language theory.

The actual origin of DNA computing can be dated in time to the summer of 1993. Its discovery so stirred the imagination that accounts of it appeared in leading newspapers and science magazines the world over. The idea for computing with DNA was the inspiration of theoretical computer scientist and mathematician Leonard Adleman of the University of Southern California as he was perusing a biology textbook. Adleman already had a highly distinguished scientific track record as co-discoverer of a way to implement public-key cryptography in the late 1970s, while a young faculty member at MIT. (Cryptography is a way to encode messages to make them unintelligible to anyone other than those authorized.) His work, a collaboration with MIT colleagues Ron Rivest and Adi Shamir, became the basis for RSA encryption, now a widely used international standard for encrypting information, especially over the Internet. It also propelled the three scientists to worldwide fame, and led them to create a successful high-tech company, RSA Inc., to handle all business matters related to RSA.

In the 1980s, Adleman had begun to study biology and learn the methods of wet lab experimentation, being attracted to the field by the unsolved problem of HIV/AIDS. It was while learning about the biomolecular mechanisms underlying in this condition that he read a section in a textbook about polymerase, the enzyme that catalyzes the elongation of a new DNA strand during DNA replication. He was struck by the similarities between the way both DNA and computers process encoded information, and it occurred to him that, by cutting, pasting, and copying DNA in the right order, it should be possible to carry out certain simple algorithms with strands of DNA. He then proceeded to astound the scientific community by actually carrying out a difficult computation, called "The Directed Hamiltonian Path Problem," on pieces of DNA in test tubes, using the standard laboratory techniques of

recombinant DNA or genetic engineering. His accomplishment generated widespread intellectual excitement among scientists because of its revolutionary nature and the seeming potential of DNA to outperform conventional computers. Some scientists saw its power in its ability to encode information in such a small space, and by manipulating the strands, to sort through an exhaustive library of all possible answers to problems of a certain size.

The Gene-Based Computer

DNA is a major constituent of the chromosomes that determine our heredity. It consists of two long molecular chains twisted into the form of a double helix, and joined by hydrogen bonds (figure 5.1). The bonds are linked to four nucleotides, adenine and thymine (A and T) or cytosine and guanine (C or G). In living organisms, the bases in each of these nucleotides bonds with its complement—A to T and C to G—in a pattern that ultimately determines our heredity. (A base is a chemical compound, distinguished from an acid, that contains oxygen and hydrogen bound together to form a hydroxyl ion, and has a pH greater than 7; a common base is sodium hydroxide, NaOH.) Two DNA strings are complementary if the second has a sequence such that A and T are interchanged, and C and G are interchanged.

In order to compute using strands of DNA, one must first start by elucidating, step by step, the computer algorithm that solves the problem of interest. Then, one must translate each of these steps into the language of DNA, by determining the sequences and exact manipulations to be performed on the DNA. The DNA starting material with the proper sequences must be custom made in a lab; these short strands are known as oligonucleotides. This process usually involves the close collaboration of computer scientists and biologists; the cross-pollination that typically results has helped foster growth in the field.

The DNA-based computation can be arranged as a series of test tubes filled with water and up to 10^{20} strands of DNA. The strands consist of the preselected sequence of oligonucleotides, which are then manipulated and sorted according to the algorithm. For example, they can be sorted by length, or by performing various "and," "or," or "not" operations to check for the presence of certain sequences in an individual strand of

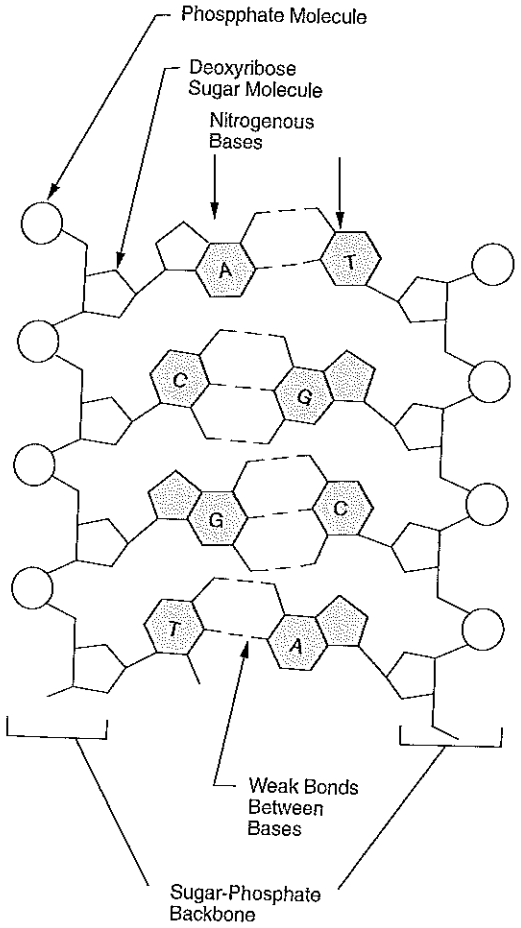

Figure 5.1
DNA structure. The four base pairs of DNA are arranged along the sugar-phosphate backbone in a particular order, encoding all genetic instructions for an organism. The bases adenine (A) pairs with thymine (T), while cytosine (C) pairs with guanine (G). The two DNA strands are held together by weak bonds between the bases. (Source: US Department of Energy Human Genome Project, <http://www.ornl.gov/hgmis>)

DNA. The strands remaining at the end of these operations are amplified using standard molecular biology lab techniques, and then are sequenced to identify the resulting DNA sequence that represents the answer to the problem (figure 5.2). In principle, only one strand containing the answer needs to be amplified.

Each operation performed sequentially on the DNA creates a new batch from the results of earlier mixtures; the operations can consist of separating the strands by length, pouring the contents of one test tube into another, extracting those strands with a given sequence, heating or cooling them, or using enzymes to splice the DNA. The series of test tube manipulations forms what computer scientists call "a single-instruction, multiple-data" (SIMD) computation performed in parallel on the DNA.

The numerous laboratory techniques of molecular biology, known as recombinant DNA techniques, enable a broad range of algorithmic operations to be used in DNA computing. These methods have been developed over the last few decades for use in biology wet labs for genetic engineering purposes. The principle laboratory operations used in DNA computing are the following:

- Synthesizing DNA strands of a desired sequence of A, T, G, or C and length.
- Separating double-stranded DNA into single strands by heating, breaking the hydrogen bonds that connect them. This is called *denaturation.*
- Fusing single-stranded DNA into double-stranded DNA with complementary base pairs connected—called *annealing or hybridization.*
- Removing all the single strands or all the double strands from the mixture.
- Cutting the strands in exact places using restriction enzymes. (An enzyme is a catalyst that causes a chemical reaction to occur.) Many bacteria produce restriction enzymes, protecting the cell by cleaving and destroying the DNA of invading viruses. Each restriction enzyme cleaves the DNA at a particular site, so each operation requires a different one.
- Separating the strands by length.
- Extracting and separating out strands with a known sequence of 15–20 base pairs anywhere along the strand.
- Finding a particular string encoded in a DNA strand.
- Constructing complementary strands for short (15–20 base pairs) strings in a strand.

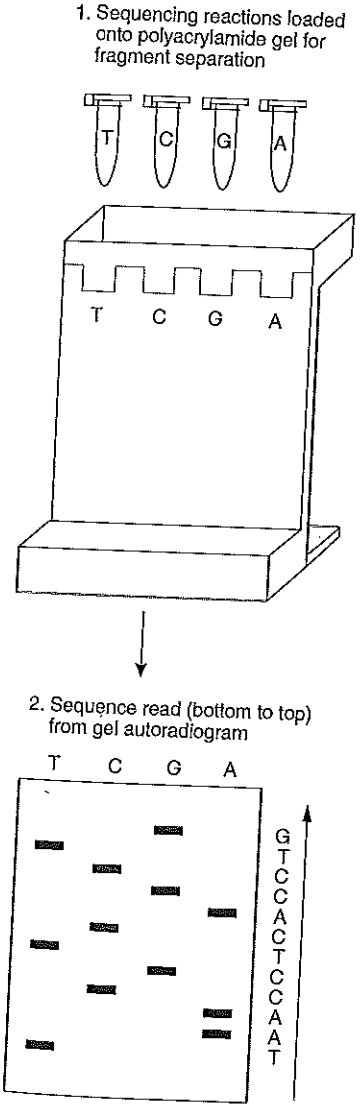

Figure 5.2
DNA sequencing. This particular method of sequencing (called the Sanger method) uses enzymes to synthesize DNA chains of varying lengths, stopping DNA replication at one of the four bases and then determining the resulting fragment lengths. The fragments are then separated by a method called electrophoresis (1) and the positions of the nucleotides analyzed to determine sequence (2). (Source: US Department of Energy Human Genome Project, <http://www.ornl.gov/hgmis>)

- Making many copies of a given strand of DNA using the polymerase chain reaction (PCR). To use PCR, double-stranded DNA is heated and separated, and then short strands that are complementary to sequences in the targeted DNA strand (primers) are added, together with free base pairs and the enzyme DNA polymerase, which catalyzes the elongation of a piece of DNA. In a series of heating and cooling cycles, the DNA doubles with each cycle.
- Appending a given string of DNA to a selected substring, or to the entire strand of DNA present, called *ligation*.
- Destroying a marked strand of DNA.
- Detecting and reading. This enables one to determine if a test tube of DNA does/doesn't contain at least one strand with the desired result; and, if it does, to interpret its sequence.

The Traveling Salesman

Adleman's first successful test of the theory of DNA computing was solving a well-known problem in computing called the Directed Hamiltonian Path Problem (or "Traveling Salesman Problem"). According to Adleman's method, a particular DNA sequence can represent the vertices or endpoints of a graph, and one can describe random paths through the graph by performing operations on the various strands of DNA. In this case, Adleman chose seven cities to represent the vertices of the graph, and fourteen links connecting the cities in various ways that represent one-way non-stop flights between two cities. The problem entails figuring out whether there is a route that takes the salesman from a given starting city to a given end city, passing through each city once and only once (figure 5.3).

Step one is to assign each city a unique six-letter name, where each letter stands for a given nucleotide (adenine, guanine, cytosine, and thymine, or A, G, C, and T). So in this case we do the following:

San Francisco = AGCTAT
New York = GTTAGC
Seattle = ACACTA
Sydney = AGTTAT
Seville = TATCTC
London = TAAAAG
Rome = GATGCT

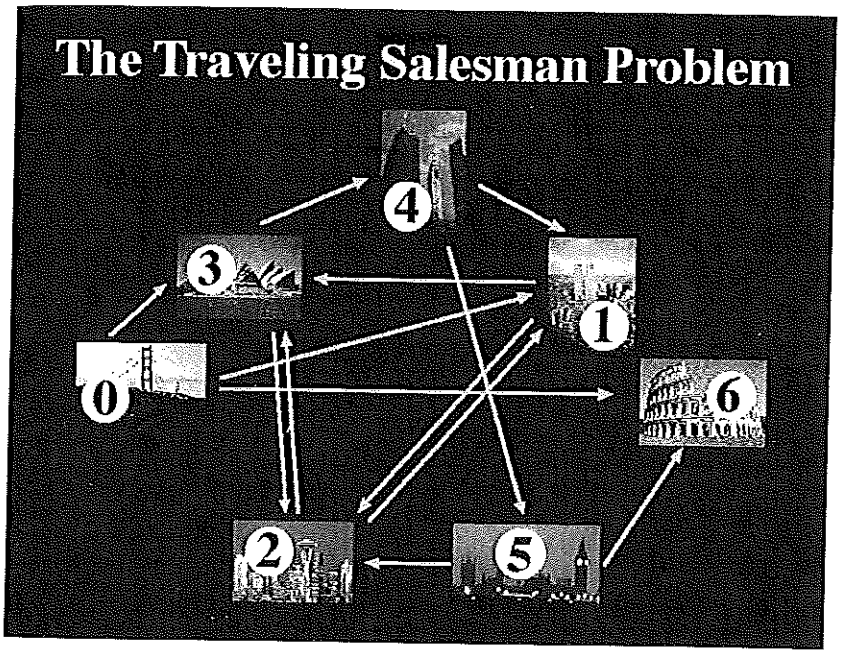

Figure 5.3
DNA computing solved the seven-city directed-path problem by reacting coded DNA strands, as represented by the letter sequences next to the pictures, to imprint a successful route on the product material. (Source: L. Kari and L.F. Landweber, "Computing with DNA," in *Bioinformatics Methods and Protocols*, S. Misener and S. Krawetz, eds., Humana, Totowa, NJ, 1999, p. 415)

Then one assigns the corresponding flight names for those cities which have direct connections, using the first three letters of the city name to signify arrival and the last to indicate departure, for example:

Departs San Francisco, arrives Sydney = TATAGT
Departs Sydney, arrives London = TATTAA

and so on. The four nucleotides only bond chemically with their complementary base pair, so each city code also possesses a complementary name, for example:

San Francisco (original code) = AGCTAT
San Francisco (complement code) = TCGATA
Sydney (original code) = AGTTAT
Sydney (complement code) = TCAATA

Single strands of DNA containing all the direct flight names and other strands containing all the complementary city names are then synthesized in a lab and mixed together in a test tube, so if a strand containing a flight from San Francisco to Sydney encounters another with the Sydney complement city name, the two will bond:

```
TATAGT
   | | |
   TCAATA
```

Then, if this strand bonds in the test tube with one containing a flight from Sydney to, say, Seville, we have:

```
TATAGTTATTAT
   | | | | | |
   TCAATA
```

and when this piece runs into another containing the Seville complement name, we get:

```
TATAGTTATTAT
   | | | | | | | | |
   TCAATAATAGAG
```

which will react with a strand for a direct connection between Seville and another city, and so on. The next step is to read out the strands containing the answer, which results from first filtering, measuring, and reading out the DNA in the test tubes so that only strands beginning with San Francisco and ending with Rome, and that contain all the five other city codes and are seven city codes long, remain in the test tube. The sequence in these strands will contain the answer.

The remarkable parallelism of the DNA computer uses brute force to try out all the possible solutions to the problem. It does so in about 10^{14} operations per second (assuming the binding of two DNA molecules is a single operation, and also that approximately 10^{14} copies of the associated oligonucleotide are added to represent each direct flight, and that half of these bind).

The traveling salesman problem forms part of a class of problems called "NP-complete," where "NP" stands for non-deterministic polynomial time. This means a set of problems whose answers can be checked in polynomial time, that is, where the computer running time needed to verify if an answer is correct or not is bounded by a polynomial func-

tion. (This is assuming that a genie has already revealed the correct answer to be checked!)

NP-complete problems, a special subset of NP problems, currently have no known polynomial time solution. What makes Adleman's accomplishment so remarkable is that by successfully demonstrating that the traveling salesman problem could be solved with DNA, he showed, as proof of concept, that the biological computer was capable of solving a small-scale NP-complete problem. (This does not mean, however, that all NP problems can be feasibly solved.)

Adleman's experiment set a completely new direction in bio-inspired computation. Even though his approach could not be scaled up indefinitely to very large combinatorial search problems as the number of individual DNA strands needed tends to grow exponentially with the size of the problem, it was a successful demonstration of a paradigm shift in computing. Moreover, it showed that individual molecules could be manipulated in the lab to implement complex algorithms—a result that those in the emerging field of nanoscience found very encouraging.

The Fledgling Field Advances

Adleman's experiment helped attract many talented scientists to the field from biology, computer science, physics, chemistry, and engineering, and growth continues in both theory and experiments with DNA computing. One type of problem that several researchers have found very amenable to DNA computations is the satisfiability problem (SAT), an important NP-complete problem in computer theory.

Solving SAT problems with DNA demonstrates how DNA computers work as powerful search engines. Satisfiability problems attempt to answer, for a given Boolean expression A, if values can be assigned to the variables in A that would make A a true statement. With DNA, any strand that encodes values that make the expression true is a potential solution. For an SAT problem with n variables, an electronic computer must test 2^n variables one by one, which can require a lot of time for large n. One theoretical approach to SAT with DNA proposed by Georgia Tech computer scientist Dick Lipton, however, can check all the variables simultaneously, providing a much more efficient method for solving the problem.

Lloyd Smith and colleagues at the University of Wisconsin have used a slightly different technique, called surface chemistry, for carrying out SAT problems with four variables, involving sixteen possible truth assignments. This method entails placing DNA that encodes all the possible Boolean variables onto surfaces, using restriction enzymes to destroy those strands that don't satisfy the Boolean formula, and reading out the results with an optical fluorescence technique.

Biochemist Kensaku Sakamoto of the University of Tokyo solved a six-variable SAT problem using "hairpin DNA," which takes advantage of DNA's tendency to get tangled and tied up in knots. Instead of adding various enzymes to destroy the DNA strands that contain the wrong answer, Sakamoto and his group designed their experiment so that, when cooled, the strands with the wrong answers would automatically fold over and form a DNA hairpin. The method, though unique, is still being refined to reduce the number of errors in the final answer.

Princeton molecular biologist Laura Landweber and colleagues solved a nine-variable SAT problem using a combination of DNA and RNA techniques. This problem is related to the "Knight's Problem" in chess and involves a possible 512 truth assignments. Len Adleman and his colleagues recently solved a twenty-variable SAT problem in the lab, after using DNA computation to exhaustively search more than a million possible answers. Performing this computation with twenty variables, the highest number reached so far, involved overcoming some difficult problems with errors. Those in the field predict that solving eighty-variable SAT problems with DNA may be the ultimate limit because at that point there is exponential growth of the volume of liquid containing the DNA. All SAT experiments require preparing a DNA "word library" beforehand, containing strands encoded with all possible truth assignments.

Another significant advance came in 1997, when Lipton and two of his graduate students devised a theoretical method for using DNA to break the data encryption standard (DES), developed by the National Security Agency. Data encryption standard makes use of a single key, among 2^{56} keys, to scramble messages. In order to break the code, one must test each of the 2^{56} keys one by one, which would take an *extremely* long time on a conventional computer. However, in the Lipton et al. approach, all the possible keys can be encoded in strands of DNA and,

through a series of steps involving extractions, replications, and other biological processing operations, they can all be searched simultaneously for the correct one. Breaking the first key would take months; however, subsequent keys can be cracked in a matter of minutes. Although Lipton's method may never actually be implemented in the lab, it does underline DNA's massive parallelism and its capacity for information storage (DNA-based computers use about a trillionth of the space that conventional computers do to store data).

Some researchers, like Duke's John Reif, have proposed using DNA as a voluminous database for information storage; it makes use of a recombinant DNA technique called the polymerase chain reaction (PCR) as a search engine. Polymerase chain reaction reproduces a fragment of DNA so as to produce many copies of the fragment. (The technique has proved an invaluable method in biotechnology, especially in forensic science, enabling amplification of minute traces of DNA for DNA fingerprinting.) Scientists at India's Central Scientific Instruments Organization have recently developed another method for DNA data storage, involving an innovative software program that encodes digital information into DNA sequences, not only for storage but also for computation.

Many of the field's proponents hope that a successful killer ap for DNA computing will ultimately be discovered, to spur the field on to the next important stage in its evolution, with additional research funding and even commercial support.

Desktop DNA?

Like any fledgling technology, especially one as novel as DNA computing, the field has its skeptics who contend that, as a new method of computation, it has claimed more than it can deliver. However, if judgments be made, a DNA-based computer should not be assessed for its usefulness as a general-purpose computer, or as a future replacement for the silicon-based PC, but only for very specialized applications. So far, DNA computing seems best adapted to small-scale search problems, and to problems that exploit its vast potential for data storage.

Other scientists—perhaps in response to some of the media hype that surrounded the field when it was new—cite the quantity of errors in the answers that stem from the difficulties in handling a DNA computer. For

instance, to read out the final result, one may want to extract only the DNA with a specified pattern from the test tube, and error rates for extraction typically are about one part in a million. Another source of error results from the decay of DNA, which particularly affects computations that take several months to do. However, reseachers are currently working to demonstrate that these error-prone computations can be rearranged so that they are virtually error free.

DNA Computing in Vivo: The Ciliate

The DNA computing described so far takes place in the lab, and can be considered an in vitro (literally, in the glassware) operation. Computing inside tiny organisms known as ciliates provides a good example of in vivo computation. Ciliates are single-celled organisms (ciliated protozoans of genus *Oxytricha* or *Stylonychia*) that live in pond water, and swim and feed with the aid of a layer of fine hairs, or cilia. They have been on earth for about two billion years. Ciliates have two different types of nuclei, the structure inside the cell containing the genetic material that determines all the cell's structures and functions. In the smaller nucleus, all the genetic material is jumbled, fragmented, and broken into smaller segments, and includes what is known as "junk DNA," which apparently serves no purpose. Scientists have discovered that some of these genes must be unscrambled in order to form the second, larger nucleus, where the genetic instructions are reorganized so that they carry out their functions properly.

Princeton molecular biologist Laura Landweber, theoretical computer scientist Grzegorz Rozenberg at the Leiden Center for Natural Computing at Leiden University in the Netherlands, and ciliate expert David Prescott at the University of Boulder, Colorado, have studied what it takes for a ciliate to reassemble a scrambled gene, how the programs are written, and how much backtracking and error-correcting the process involves. Rozenberg calls this new computational paradigm "computing by folding and recombination." He has used elaborate constructs to help discern the logical rules followed by the ciliate in untangling its genes. Understanding this process might help scientists grasp how random molecules in the primordial soup eventually organized themselves into the genetic code.

Whither DNA Computers?

In the final analysis, there's always a risk in trying to predict how any particular scientific development will fare in the future and what its impact will be, particularly something as unique as DNA computing. However, according to originator Len Adleman, its ultimate success or failure even as a special purpose computer may even not matter. He observes that

> whether or not DNA computers will ever become stand alone competitors for electronic computers—which is unlikely—is not the point. Every living cell is filled with thousands of incredibly small, amazingly precise instruments in the form of molecules, which comprise "Nature's Tool Chest," a tool chest for the 21st century. These molecules store information, store energy, act like motors, or like structural material, and can cut and paste. Each is incredibly small, extraordinary precise, and functions with an energy efficiency that is on the cusp of what is thermodynamically feasible. I believe things like DNA computing, along with the other ways we are learning to use these wonderful tools inside the cell, will eventually lead the way to a "molecular revolution," which ultimately will have a very dramatic effect on the world.[1]